論文 / 著書情報
Article / Book Information

Title	Comparison of Replica Assisting and Speed Adjustment for Service-awareHorizontal Data Migration on Autonomous Disks
Author	Dai Kobayashi, Haruo Yokota
Journal/Book name	Proceedings of the Second IEEE International Conference on Digital Information Management, , , pp. 539-544
Issue date	2007, 10
DOI	http://dx.doi.org/10.1109/ICDIM.2007.4444279
URL	http://www.ieee.org/index.html
Copyright	(c)2007 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other users, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works for resale or redistribution to servers or lists, or reuse of any copyrighted components of this work in other works.
Note	このファイルは著者（最終）版です。 This file is author (final) version.

Comparison of Replica Assisting and Speed Adjustment for Service-aware Horizontal Data Migration on Autonomous Disks

Dai Kobayashi
Tokyo Institute of Technology
Japan Society for the Promotion of Science
daik@de.cs.titech.ac.jp

Haruo Yokota
Tokyo Institute of Technology
yokota@cs.titech.ac.jp

Abstract

We have previously proposed Autonomous Disks System as a dependable and high-performance networked parallel storage system. The system has function of load balancing by online horizontal data migration. However, the migration causes a reduction of system availability by insensitively utilizing of system resources for massive data transfers. Thus we have also proposed Replica-assisted Migration, which solve the problem by using replica data for data redundancy. In this paper we show the efficiency of this method with experimental results on simulation program, comparing to speed adjusting method which is widely used for service-aware data migration.

1 Introduction

We have previously proposed Autonomous Disks as a dependable and high-performance networked parallel storage system. The system comprises many intelligent storage nodes and high-speed local network. The node contains not only hard disk drive(s) but also cache memory and CPU. Data is distributed, replicated, and stored in each storage node, and is partitioned into short segments such as files, extents, pages, or blocks.

The system has function of load balancing by online horizontal data migration. Horizontal data migration is a method to balance workloads for removing hot spots in which heavy load is concentrate on parallel storage systems and distributed databases [4, 11]. Transferring data from hot spots to unused storage nodes, the system keeps performance of accesses on all data even when workload trend changes.

However, data migration causes temporary performance degradation since data transfer often consumes system resources such as I/O bandwidth and network bandwidth [3, 9, 14]. The storage nodes with heavy accesses need to serve accesses of both services and data migration, although the storage node decreases its performance extremely with excessive load. Saturated I/Os such as disks or networks of partial storage nodes make a system difficult to guarantee its performance. On the other hand, data migration for load balancing should be performed quickly because slow migration which cannot follow changes in access trends, results in access concentrations that the system should have avoided. Therefore, using the sensitive speed adjustment which is commonly used for data migration in hierarchical, or vertical, storage [3, 9, 14] is discouraged.

We have also proposed adaptive Replica-assisted Migration (RM), which reduces and distributes the load caused by data migration by using replica data for data redundancy [6]. The method reduces the amount of transferred data and provides a source node selection by using replica data location. The method also has ability to keep performance of each storage node by forwarding moderate ratio of accesses to replicas. A notable point of the method is that the method needs not to decrease data migration speed.

In this paper we show the efficiency of the method with experimental result on simulation program, comparing to speed adjust method which is widely used for service-aware data migration. We first evaluate the methods with Zipf-based synthetic workloads. The results show that the proposed method indicates performance gain and the gain is related to migration duration. We also use Web-based real workload for evaluating applicability of the methods. The results also show the ascendancy of the proposed method. It is because the migration speed with proposed method is faster than speed of workload trend evolution, while that with speed-adjust or normal migration lose against workload trend.

The remainder of this paper is organized as follows. We describe the Autonomous Disks System in Section 2. In Section 3, we summarize the RM migration we have previously proposed. In Section 4, we describe the experimental environment. Sections 5 and 6 give experimental results and discuss using the simulation program with a synthetic

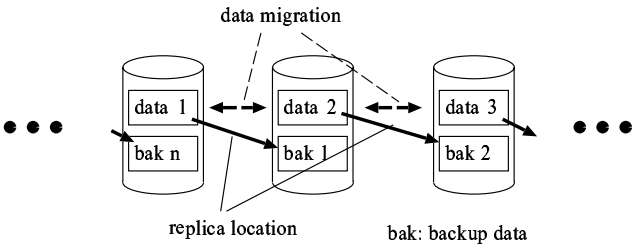

Figure 1. Chained Declustering and range partitioning

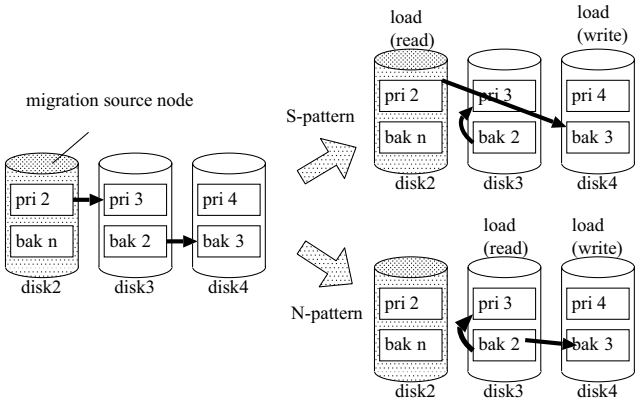

Figure 2. Two data transfer source nodes. The coordinator node can select the source node from pri$_i$ on disk$_i$ or bak$_i$ on disk$_{i+1}$ when migrating data$_i$

workload and a real workload, respectively. The final section contains our conclusions.

2 Autonomous Disks

We have previously proposed Autonomous Disks [12]as dependable and high-performance networked parallel storage systems. This adopts Primary-backup one-copy replicas [2] and the Chained Declustering replica-placement strategy [5], in which each node has replicas of one neighbor node and locates a replica of its stored data in another neighbor to achieve greater reliability. It utilizes Fat-Btree [13], which is a kind of distributed B^+-Tree with no single control point, as the storage virtualization structure. It also utilizes a range partitioning data distribution strategy, whereby each storage node contains data with an assigned field range. Items of stored data have unique identifiers and are sorted by the identifier. Each node has a range of identifiers. data are stored in the node that has range corresponding to the identifier. Figure 1 shows the combination of Chained Declustering and range partitioning.

The Autonomous Disks system has function for load balancing by data migration that is executed by communication of intelligent controllers on each storage nodes. All intelligent storage nodes on the system record degree of current load, and the planning task to decide a migration strategy is executed on a *coordinator* node, which aggregates the current workload information for all nodes. Storage nodes execute the following tasks at regular predefined time intervals. Firstly, a node is picked out to be coordinator and each node send its current load information to the coordinator. Then the coordinator node calculates the average load degree and creates a migration quantity list that includes information for each node on how much data or load should migrate to another node. To complete the strategy, each storage node transfers the specified quantity of data to the specified nodes.

3 Replica-assisted migration

We use replica data for data redundancy for realizing data migration and access quality simultaneously. We first reduce the quantity of transferred data and select the migration source node by using replica location. We also use replicas for access services, by forwarding partial accesses of primary data to replicas. An advantage of our method is that the system can migrate a substantial quantity of data even if the source node has already become a hot spot without decrease migration speed.

3.1 Migration source selection using replicas

Multiple replica locations make selection of the migration source node possible, which enables allocation of migration load to less accessed nodes.

The migration destination node is restricted to either the *right* neighbor node, which stores data with higher identifiers, or the *left* neighbor node, which has lower identifiers. Therefore, using replicas on the right node, the migration can select the data transfer source node as either primary data or replica data. We call the former the *S(Self)-pattern* and the latter the *N(Neighbor)-pattern*. Figure 2 shows the S- and N-patterns of right migration.

When data migration is required, the coordinator can select the colder migration source node from the S- or N-pattern using aggregated workload information.

3.2 Access forwarding to replicas

Partial read accesses for data stored in a hot spot node that is also a source node for data migration can be forwarded to nodes that store replicas of the primary data to

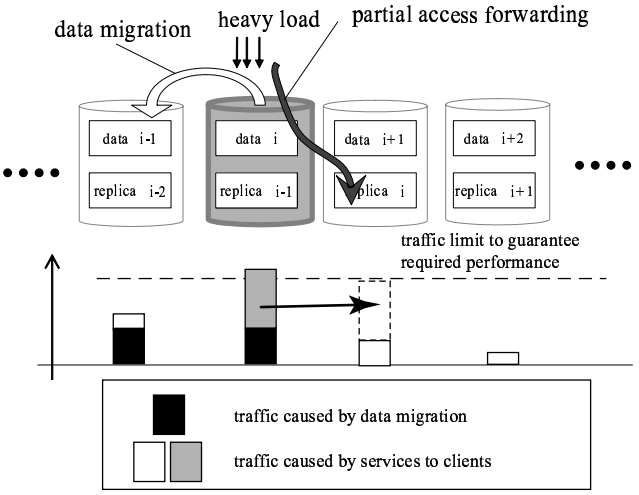

Figure 3. A concept of access forwarding for migration load

avoid performance degradation caused by the data migration. Therefore, data replacement can be executed even if a hot spot has appeared. Figure 3 illustrates the concept of the partial access forwarding.

The number of read accesses forwarded to replicas is important to provide stable data transfer services to both clients and data migration processes. In other words, the method calculates the ratio that maximizes the resources for data migration at the data migrating node while staying within the maximum storage node performance limits. The limiting load values for maintaining tolerable availability of a system are estimated before running the system, from service level agreements, specifications of disks, networks, and the brief characteristics of workload trends on nodes, such as cache hit ratios and request sizes.

The access forwarding ratio r_i on node i is calculated at the coordinator node from aggregated current workload information, which includes the load quantity and the write request ratio, and is communicated to node i . Our method executes the following tasks in addition to the regular migration tasks described in Section 2. Firstly, the coordinator calculates the access forwarding ratios r_i, as well as calculating the quantity of migrating load, using the load degree aggregated from all nodes. Secondly, the coordinator sends r_i to node i for all i and the nodes each apply their own r_i. Thirdly, nodes migrate stored data to other nodes, as required. At this point, read accesses to the nodes are partially forwarded to the nodes that store replicas according to each forwarding ratio r_i. When each node has finished migrating the specified quantity of data, it sets its r_i to zero. Detailed information of algorithm for calculating r_i is described in [6].

Table 1. Simulation Parameters

Simulation Parameter	Value
Rotational speed (RPM)	7200
# of surfaces	6
# of sectors per cylinder	4320 – 9000
Seek time (msec)	0.8 – 15.1
Buffer size (KB)	8196
Disk node	4
Cache size (MB)	512
Network bandwidth (Mbps)	800

Access forwarding to nodes storing replicas decreases the hit ratios of cache memory on each storage node because both the primary and replica data of other nodes may be stored in the same cache and will compete for cache space. In this paper, we use a method for choosing forwarded accesses that uses the hit ratios of each cache to estimate the popularity of data at low cost [7]. It helps the replication to use the limited cache space efficiently if accesses for data in less demand are forwarded. In addition, it also strictly distinguishes between primary accessed data and replica accessed data to avoid competition for cache space.

4 Experiment Environment

We observed and evaluated the behavior of our method with the Autonomous Disks System, using a Zipf-based synthetic workload and a file server workload with a simulation program. In this section, we describe the simulation environment and the workloads. The results of experiments with the synthetic workload are given in the next section and those with the real workload are given in Section 6. We used a queuing network-based storage system simulation program. The behavior of the hard disk drives in the simulation was calculated with parameters from an HGST Deskstar T7K500 specification [10]. Table 1 gives the simulation parameters.

We implemented the data migration strategy described in Section 2. The load value was defined as the summation of request sizes in the previous 600 seconds. The sizes were modified by using the storage node performance observed in preparatory experiments, giving the smaller size a relatively larger load value per unit of size.

In each experiment, we compared three methods: data migration without performance-aware control (NORMAL), RM method, and the speed adjust method. We implemented Aqueduct [9] as speed adjust method. It adjust migration speed adaptively by using observed response time on each node, respectively.

Table 2. Specification of experiments with synthetic workload

Workload parameter	Value
Time span (hours)	2
Total file size (GB)	100×2 (Primary & Backup)
# of files in total	100,000 (1MB each)
Access distribution	Zipf $\left(f(k;s,N) = \frac{1/k^s}{\sum_{n=1}^{N} 1/n^s}\right)$
Zipf exponent s	1.5, 1.2

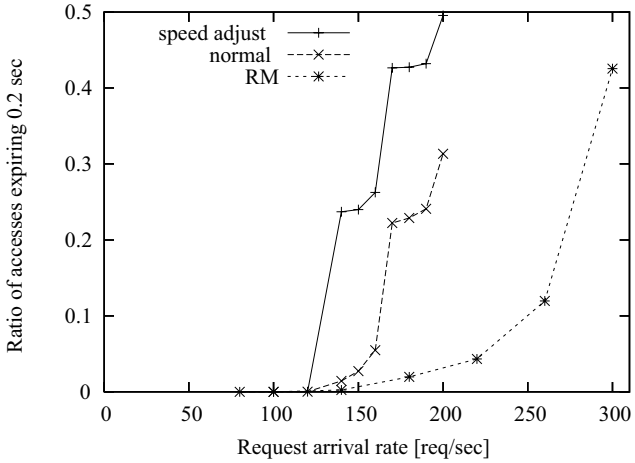

Figure 4. The ratio of accesses expiring within 0.2 sec(Zipf parameter s=1.5)

5 Synthetic data

First, we observed the behavior of each method with a synthetic workload. The workload specification is given in Table 2. The workload was generated by using a Zipf-based access distribution and a Poisson arrival distribution. The Zipf distribution is described in the table, where f is the ratio of accesses to a file ranking k in N files and s is a parameter indicating the degree of skew. We set the workload trend at 10 minutes after the start of the simulation by changing the seed of a random value generator for a shuffle function that decided the target file for each access, followed by migration start at 5 minutes after the workload change.

Figure 4 shows the number of access requests expiring within 0.2 seconds, as an agreed service level. The 0.2-second value is about ten times the latency under low load. The horizontal axis indicates scale of workload quantity. While the requests for NORMAL and speed adjust data migration increased sharply, those for the two RM methods increased at a slower pace, even for heavier workloads.

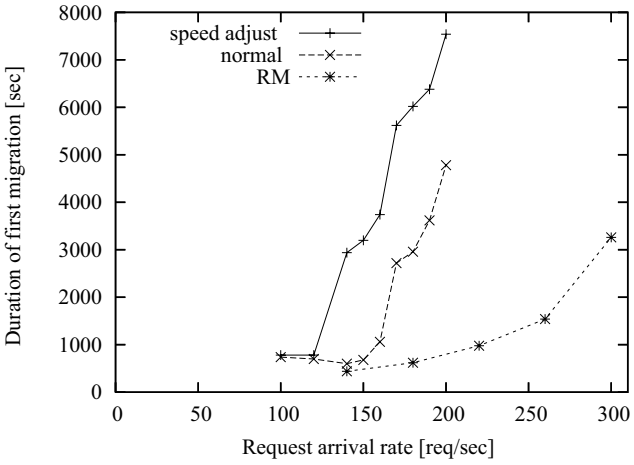

Figure 5. Duration of first data migration (Zipf parameter s=1.5)

The performance degradation is well related to a data migration duration. Figure 5 shows duration from when the first data migration occurred to when the migration finish to transfer required quantities of data in same experiments as Figure 4. The figure 5 shows that the duration with both speed adjust method and normal migration increases extremely when request arrival rate increases. It is a result of that data migration and accesses for storage services compete for resources on hot spots. On the other hand, that of the RM method increases slower than the others. It is effect of access forwarding and migration source node selection that move traffic of migration to replica nodes that has enough resources.

When degree of access distribution is less sharp, the effect of RM becomes smaller because usable resources on replica nodes relatively small. It is observed in figure 6 that is the number of expiring accesses when Zipf parameter s set to 1.2 that is less sharp than $s = 1.5$. Although the figure shows a same characteristic as figure 4, performance gain by RM method is 33% that is less than 38 % on the experiment with $s = 1.5$.

6 Web server data

In this section, we evaluate the efficiency of methods by using access trace logs of large scale web server as more real workload. Actual workloads is much complex than that of previous because it has quick and large evolution, or concept drift, and also has various size of requests. The properties lead to an increase of error in estimated value of load quantities. We use a workload generated by partial terms of access logs of FIFA WorldCup98 Official Web [8].

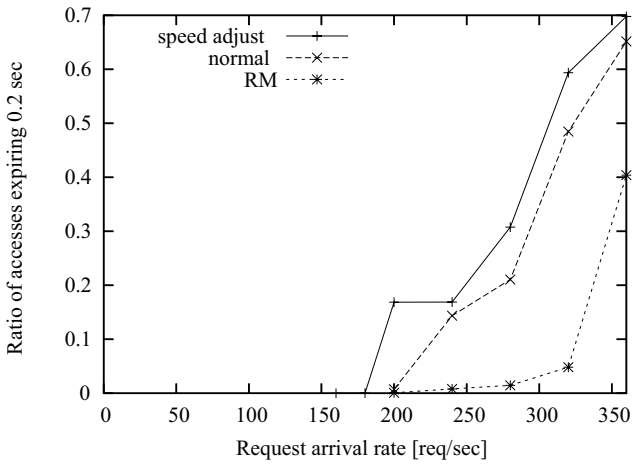

Figure 6. The ratio of accesses expiring within 0.2 sec(Zipf parameter s=1.2)

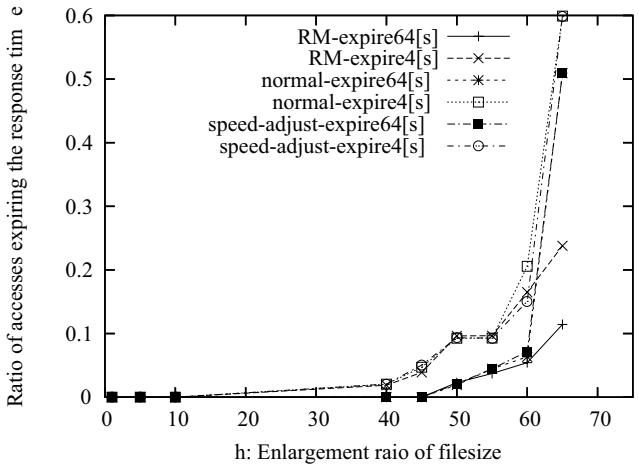

Figure 7. The ratio of accesses that expiring within four second and 64 second with Web-based workload

Table 3. Specification of experiments with real server workload

workload parameter	value
time span	2 hours
# of requests	5,000,000
read:write	10:0
total file size	430MB $\times h\times$ 2 (Primary & Backup)
# of total files	21000
accessed file size	277MB $\times h$
# of accessed files	6300

The workload specification is given in Table 3. More information of the workload is on [1]. The period on workload used in the experiment has two major evolution of access trend. Thus the experiment consists two large load balancing; from initial state to stationary state, and from stationary state to state after trend evolution. Results of NORMAL and RM include two migrations. However, that of speed-adjust method includes only latter migration because speed-adjust stop migration tasks if speed-adjust is active from initial state, or extremely skewed state. Request sizes of the workload in each experiment multiplied by h which is defined as a load parameter for expressing various quantities of workload.

Figure 7 shows the ratio of the number of access requests expiring within 4 seconds and 64 seconds, as an agreed service level, for the experiments with various load parameter h. In the figure, the horizontal axis indicates the load parameter h.

The result shows the efficiency of the RM method with real workload clearly. Contract to the result of synthetic one, all method indicates same performance on the result with h lower than 60. It is because there are few access concentration that decrease node performance on the experiments. Thus, speed-adjust method hardly decrease migration speed on it. However, speed-adjust and normal method indicates significant performance degradation when h equals 65. Analytic result of simulation log shows that the degradation was caused by data location change slower than access trend change. The situation that data migration and accesses for storage services compete for resources on hot spots results in the decrease of the migration speed. On the other hand, RM method can execute data migration quickly because the method uses resources on replica node.

The results indicate that the important things for horizontal migration and performance keeping is quick data migration. The proposed method have ability to realize it and the speed adjust method does not.

7 Conclusions

In this paper, we analyzed and evaluated Replica-assisted migration we have proposed, focusing on migration duration. The method reduces the amount of data transferred and provides a source node selection by using replica data location. The method also achieves to keep performance of each storage node by forwarding moderate ratio of accesses to replicas.

We first experimented with a Zipf-based synthetic workload using several parameter values, and showed that the method can maintain availability and migrate required data

quickly. The second experiment used a workload generated by real large-scale Web server trace logs, and showed that the method can work better with practical workload trends that with speed-adjust method. The two experimental results shows that it is significant for keeping service performance with horizontal data migration to transfer required quantity of data as quick as possible.

Our current issues are treatment of write requests. Our previous work [6] shows that one of our method to provide migration source node selection worked well with the workload, while the other of our method to forward partial accesses to replicas made no sense because of high skew access distribution and high update load. We consider using the concept of neighbor WAL method for eliminating quantities of load on replicas caused by write operations.

Another future work will include detailed analysis and evaluation of the method under various workloads with different characteristics, such as OLAP, video servers, and scientific applications. Analysis will be required not only of the method itself but also of underlying techniques such as the migration strategy and the workload estimation method.

8 Acknowledgments

This work was partially supported by Japan Science and Technology agency via CREST program, Japan Society for the Promotion of Science via Grant-in-aid for JSPS Fellows, Storage Research Consortium, MEXT of Japanese Government via Grant-in-Aid for Scientific Research on Priority Areas, #18049026 and the MEXT 21st Century COE Program.

References

[1] M. Arlitt and T. Jin. Workload characterization of the 1998 world cup web site. Technical Report HPL-1999-35R1, Hewlett-Packart Laboratories, Oct. 1999.

[2] P. A. Bernstein and N. Goodman. An algorithm for concurrency control and recovery in replicated distributed databases. *ACM Trans. Database Syst.*, 9(4):596–615, Dec. 1984.

[3] K. Dasgupta, S. Ghosal, R. Jain, U. Sharma, and A. Verma. Qosmig: Adaptive rate-controlled migration of bulk data in storage systems. In *the 21st International Conference on Data Engineering (ICDE2005)*, pages 816–827, 2005.

[4] H. Feelifl, M. Kitsuregawa, and B. Ooi. A fast convergence technique for online heat-balancing of btree indexed database over shared-nothing parallel systems. In *11th Int'l Conf. on Database and Expert Systems Applications*, pages 846–858, Sep 2000.

[5] H.-I. Hsiao and D. J. DeWitt. Chained declustering: A new availability strategy for multiprocessor database machines. In *Proceedings of the Sixth International Conference on Data Engineering*, pages 456–465, Los Angeles, CA, Feb. 1990. IEEE Computer Society.

[6] D. Kobayashi, R. Taguchi, and H. Yokota. An experimental evaluation of the adaptive replica-assisted migration for parallel storage systems. In *The Third IEEE International Workshop on Databases for Next-Generation Researchers (SWOD 2007)*, 2007.

[7] D. Kobayashi, A. Watanabe, R. Taguchi, T. Uehara, and H. Yokota. An efficient access forwarding method based on caches on storage nodes. In *International Special Workshop on Databases For Next Generation Researchers (SWOD 2005) In Memoriam of Prof. Kambayashi*, pages 188–191, 2005.

[8] L. B. N. Laboratory. The internet traffic archive. http://-ita.ee.lbl.gov/.

[9] C. Lu, G. A. Alvarez, and J. Wilkes. *Aqueduct:* online data migration with performance guarantees. In *Conference on File and Storage Technologies (FAST'02)*, pages 219–230, Monterey, CA, Jan. 2002.

[10] H. G. S. Technologies. *Deskstar T7K500 Hard Disk Drive Specification*. http://www.hitachigst.com, ver. 1.2 edition, 2006.

[11] G. Weikum, A. Mönkeberg, C. Hasse, and P. Zabback. Self-tuning database technology and information services: from wishful thinking to viable engineering. In *VLDB*, pages 20–31, 2002.

[12] H. Yokota. Autonomous Disks for Advanced Database Applications. In *Proc. of International Symposium on Database Applications in Non-Traditional Environments (DANTE'99)*, pages 441–448, Nov. 1999.

[13] H. Yokota, Y. Kanemasa, and J. Miyazaki. Fat-Btree: An Update-Conscious Parallel Directory Structure. In *Proc. of the 15th Int'l Conf. on Data Engineering*, pages 448–457, 1999.

[14] J. Zhang, P. Sarkar, and A. Sivasubramaniam. Achieving completion time guarantees in an opportunistic data migration scheme. *SIGMETRICS Perform. Eval. Rev.*, 33(4):11–16, 2006.